

POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Matematica

Tesi di Laurea Magistrale

**Automatizzazione della tecnica di analisi delle
componenti principali sparse caratterizzanti
una collezione di documenti testuali**



Relatore:

Prof.ssa Tania Cerquitelli

Correlatore:

Dott.ssa Evelina di Corso

Candidata:

Sara Verbaro

Anno Accademico 2018/2019

Abstract

Con il passare degli anni i progressi nella tecnologia hardware e software hanno portato ad un costante aumento della quantità di dati in circolazione. In particolare con l'aumento di dati testuali, generati e raccolti dai social network alle biblioteche digitali, hanno avuto sempre più importanza processi come il text mining e il topic modeling che studiano algoritmi in grado di estrarre l'informazione utile da un'ampia collezione di documenti. Questo tipo di attività comporta più fasi e richiede la scelta di parametri specifici per ogni algoritmo proposto: diventa perciò necessaria la presenza di esperti e di analisti in grado di guidare il processo di recupero delle informazioni nascoste. Affinchè il processo di mining sia efficiente si ha l'esigenza di renderlo automatico, senza il bisogno di una continua visione umana dell'attività.

Questa tesi mira ad implementare un approccio proposto da Zou et al. in grado di raggruppare automaticamente i documenti presenti in una collezione di dati testuali in gruppi coerenti e ben separati attraverso una metodologia di topic modeling. La tecnica proposta in questo elaborato è l'analisi delle componenti principali sparse la quale, dati il numero di topics e il parametro di sparsità, ci consente di individuare documenti che appartengono allo stesso topic e per ogni topic ci fornisce una lista di parole che meglio lo descrivono. Per supportare l'analista nella corretta selezione dei parametri sono stati proposti due approcci.

L'algoritmo è stato sviluppato in Python e convalidato su diverse raccolte di dati reali, con caratteristiche e proprietà testuali diverse. Dal risultato sperimentale ottenuto, il tuning dei parametri supporta in maniera automatica

l'analista. Inoltre, l'algoritmo è efficiente nel descrivere e raggruppare le collezioni di documenti testuali forniti ed è efficace nel semplificare il processo di analisi.

Indice

Introduzione	7
1 Stato dell'arte	9
Stato dell'arte	9
2 Sparse PCA	13
Sparse PCA	13
2.1 Origini	13
2.2 Formulazione dell'algoritmo	15
3 Metodologia	23
Metodologia	23
3.1 <i>Pre-processing</i>	23
3.2 Modellazione del documento	26
3.3 <i>Sparse PCA</i>	27
3.4 Scelta dei parametri	28
3.5 Clustering	31
3.6 Visualizzazione	33
3.6.1 Rappresentazione t-SNE	33
3.6.2 Rappresentazione nube di parole	36
4 Risultati	37

Risultati	37
4.1 Collezioni Wikipedia	38
4.2 Collezione di Tweet	45
Conclusione e sviluppi futuri	49
Bibliografia	51

Introduzione

Con il passare degli anni i progressi nella tecnologia hardware e software hanno portato ad un costante aumento della quantità di dati in circolazione. In particolare con l'aumento di dati testuali, generati e raccolti dai social network alle biblioteche digitali, hanno avuto sempre più importanza processi come il *text mining* e il *topic modeling* che studiano algoritmi in grado di estrarre l'informazione utile da un'ampia collezione di documenti. Questo tipo di attività comporta più fasi e richiede la scelta di parametri specifici per ogni algoritmo proposto: diventa perciò necessaria la presenza di esperti e di analisti in grado di guidare il processo di recupero delle informazioni nascoste. Affinchè il processo di *mining* sia efficiente si ha l'esigenza di renderlo automatico, senza il bisogno di una continua visione umana dell'attività.

Questa tesi mira ad implementare un approccio proposto da Zou et al. in grado di raggruppare automaticamente i documenti presenti in una collezione di dati testuali in gruppi coerenti e ben separati attraverso una metodologia di *topic modeling*. La tecnica proposta in questo elaborato è l'analisi delle componenti principali sparse la quale, dati il numero di topic e il parametro di sparsità, ci consente di individuare documenti che appartengono allo stesso topic e per ogni topic ci fornisce una lista di parole che meglio lo descrivono. Il lavoro di tesi prevede lo sviluppo di un framework caratterizzato da tre blocchi principali. Il primo comprende diverse operazioni di *pre-processing* e di modellazione dei documenti presenti nel corpo, e ci consente di estrarre una matrice, indicata con "*TF - IDF*". Il secondo blocco, invece, fornisce un'analisi automatica dei parametri di input dell'algoritmo di *Sparse PCA* e

la sua successiva implementazione a partire dalla matrice $TF - IDF$. Infine l'ultimo blocco prevede l'implementazione dell'algoritmo di *K-Means Clustering*, che ci consentirà di raggruppare i documenti presenti nella collezione in base al topic che li caratterizza, e fornisce alcune visualizzazioni, tra cui grafici t-SNE, *scatter plot* e *word cloud*, in modo da analizzare i risultati ottenuti.

L'algoritmo è stato sviluppato in Python e convalidato su tre raccolte di dati reali, con caratteristiche e proprietà testuali diverse: due collezioni di dati testuali importati da Wikipedia e una collezione di Tweet. Dai risultati sperimentali ottenuti, il *tuning* dei parametri supporta in maniera automatica l'analista. L'algoritmo è in grado di raggruppare automaticamente i documenti presenti in una collezione di dati testuali in gruppi coerenti e ben separati e di fornire, per ogni topic, la lista dei termini che lo caratterizzano. L'elaborato presenta la seguente struttura. Il capitolo 1 fa riferimento allo stato dell'arte del metodo: viene fornita una descrizione del modello probabilistico LDA e della tecnica proposta da Christophe Croux, Peter Filzmoser e Heinrich Fritz, in grado di fissare automaticamente il numero ottimale di topic e il parametro di sparsità per l'algoritmo di *Sparse PCA*. Il capitolo 2 introduce la tecnica di *Sparse PCA*, presentandone due diverse formulazioni. Il capitolo 3 fornisce l'architettura del modello, analizzando nel dettaglio i tre blocchi del framework proposto. Il capitolo 4, invece, illustra le performance del modello su tre collezioni di dati testuali. Infine, nel capitolo finale, vengono riportate le conclusioni e gli eventuali sviluppi futuri del lavoro.

Capitolo 1

Stato dell'arte

L'algoritmo comunemente usato per l'estrazione dei topic in una collezione di documenti è l'LDA (Latent Dirichlet Allocation) ¹. Questo modello consente di esprimere il risultato come una distribuzione di probabilità su tutti i termini: solo i valori con il peso più alto vengono considerati, i rimanenti sono scartati. Tuttavia il fattore di sparsità non viene incorporato direttamente nel modello, bensì richiede un procedimento finale di thresholding che consente di rendere il risultato più interpretabile.

La sparsità è una componente cruciale soprattutto nel caso di una collezione di documenti testuali in quanto ha un grosso impatto nel renderne efficiente la sintesi, il confronto e la visualizzazione dei dati. Pertanto sarà di nostro interesse presentare una nuova tecnica che tenga conto di questo fattore direttamente nell'implementazione dell'algoritmo: introduciamo così il modello di Sparse PCA (*Sparse Principal Component Analysis*) ², basato sull'analisi delle componenti principali sparse. Vedremo come le soluzioni saranno più veloci da calcolare e soprattutto consentiranno una maggiore interpretabilità del risultato. Anche per questo algoritmo, come nel caso dell'LDA, bisogna prestare particolare attenzione nel selezionare il numero ottimale di topic, indicato con K . Inoltre vi è un altro parametro che bisogna tenere in conto:

¹Blei et al., "*Latent Dirichlet Allocation*" (2003)

²Zou et al., "*Sparse Principal Component Analysis*" (2006)

il parametro di sparsità, che indicheremo con α .

Christophe Croux, Peter Filzmoser e Heinrich Fritz ³ proposero una nuova tecnica per la selezione di questi due parametri. Per prima cosa raffigurano in uno *scree-plot* la percentuale di varianza spiegata dalle componenti principali rispetto al numero di componenti principali stesse: in questo modo la scelta di K , cioè del numero di componenti principali, viene fatta in base al grado di percentuale di varianza che si intende raggiungere (un esempio è riportato in figura 1.1). Per ottenere questo risultato è stato implementato l'algoritmo di PCA standard, che equivale a quello di Sparse PCA calcolato per un valore del parametro di sparsità pari a zero. Una volta individuato il

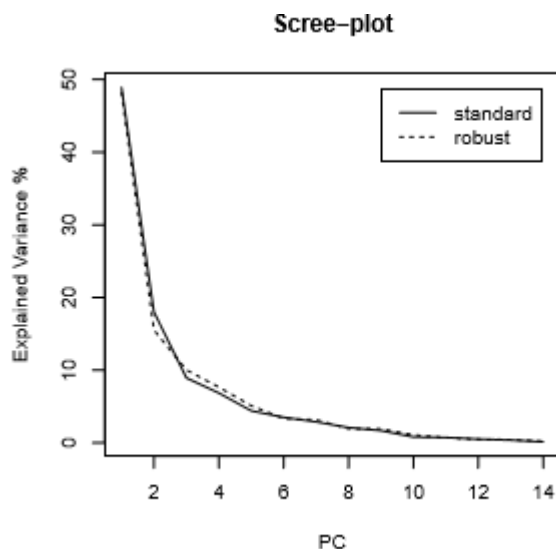


Figura 1.1: Scelta del parametro K

parametro K viene presentata una tecnica in grado di selezionare il parametro di sparsità, nel loro articolo indicato con λ . Propongono di selezionare quel valore di λ che sia in grado di minimizzare l'indice BIC, dall'inglese Bayesian Information Criterion ⁴. L'indicatore da loro proposto presenta la

³Christophe Croux, Peter Filzmoser e Heinrich Fritz, *"Robust Sparse Principal Component Analysis"* (2013)

⁴Guo et al. (2010), Leng e Wang (2009)

seguinte forma:

$$BIC(\lambda) = \frac{\hat{R}V}{RV} + df(\lambda) \frac{\log(n)}{n},$$

dove i valori $\hat{R}V$ e RV fanno riferimento alla varianza totale della matrice dei residui, ottenuta rispettivamente dalla Sparse PCA e dalla PCA standard con K componenti principali. Questi ultimi valori vengono calcolati nel seguente modo:

$$\hat{R}V = \nu(X - X\hat{A}_K\hat{A}_K^t)$$

$$RV = \nu(X - XA_KA_K^t),$$

dove X indica la matrice dei dati, mentre $\hat{A}_K$ e A_K rappresentano le matrici dei caricamenti ("loadings") contenenti le prime K componenti principali, rispettivamente della Sparse PCA e della PCA standard. Precisiamo che ν indica la varianza totale di una matrice ed è definita come:

$$\nu(Y) = \sum_{s=1}^p \nu(y_s),$$

dove Y è una qualsiasi matrice di dimensione $[n \times p]$.

D'altra parte il termine $df(\lambda)$ indica il numero di caricamenti diversi da zero dal momento in cui viene usato λ come parametro di penalità. Di conseguenza il primo termine nella formula BIC misura la qualità della forma, mentre il secondo termine penalizza la complessità del modello. Per cui la tecnica proposta per la selezione del parametro λ consiste nella minimizzazione dell'indice di BIC su un range $[0, \lambda_{max}]$, dove λ_{max} indica una sparsità totale.

Inoltre in questo articolo viene presentato uno strumento grafico che, in aggiunta al BIC, consente di selezionare un valore opportuno di λ : la cosiddetta curva trade-off. In questa rappresentazione viene raffigurata la percentuale di varianza spiegata dalle K componenti principali sparse in funzione di λ , come si può osservare dall'esempio fornito in figura 1.2 . Il grafico mostra come all'aumentare del parametro di sparsità si ha una diminuzione della varianza spiegata. L'idea da loro proposta consiste nel selezionare un valore di λ prima che si abbia un brusco declino della curva, nell'esempio mostrato in figura hanno optato per il valore indicato dalla linea rossa tratteggiata. Il

valore ottenuto dalla curva trade-off dovrebbe corrispondere a quello calcolato usando il criterio di BIC.

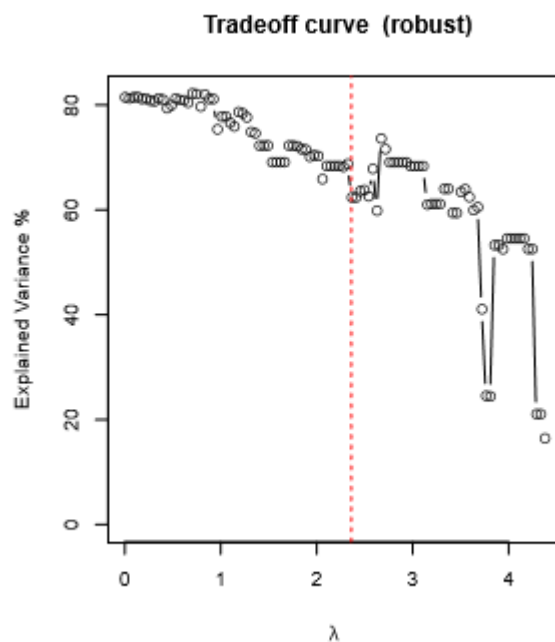


Figura 1.2: Scelta del parametro λ

Questo elaborato propone un nuovo metodo per fissare automaticamente il numero ottimale di topic K e il parametro di sparsità, che nel nostro caso verrà indicato con α , in modo da analizzare una collezione di documenti attraverso l'algoritmo di *Sparse PCA*.

Capitolo 2

Sparse PCA

In questo paragrafo verrà presentata la metodologia di *Sparse PCA* che ci permetterà di raggruppare automaticamente i documenti presenti in una collezione di dati testuali in base al topic che li caratterizza. In particolare, dopo un breve cenno sul contesto in cui è nata e si è sviluppata, analizzeremo due formulazioni del modello. In questo modo sarà più facile capire il lavoro svolto dagli algoritmi trattato nella sezione successiva.

2.1 Origini

L'analisi delle componenti principali (in inglese Principal Component Analysis, abbreviata PCA) rappresenta una delle tecniche più importanti nell'ambito dell'analisi multivariata in quanto consente di pre-elaborare un'elevata quantità di dati, di ridurre la dimensionalità e di estrarne le caratteristiche principali. Questo metodo, proposto per la prima volta nel 1901 da Karl Pearson e sviluppato poi da Harold Hotelling nel 1933, consente di ridurre il numero di variabili che descrivono un insieme di dati a un numero minore di variabili che prendono il nome di componenti principali (in inglese Principal Component, PCs). L'algoritmo di PCA è caratterizzato da due diversi tipi di decomposizione, a seconda della struttura della matrice di partenza: la decomposizione a valore singolare e la decomposizione degli autovalori. La

prima viene comunemente chiamata "SVD" (dall'inglese Singular Value Decomposition) e lavora a partire dalla matrice dei dati, indicata con X . In questo caso le variabili originarie vengono proiettate in un nuovo sistema cartesiano: viene fatta una trasformazione lineare delle variabili, in particolare la nuova variabile con maggiore varianza viene proiettata sul primo asse, la successiva sul secondo asse, e così via. La decomposizione degli autovalori, invece, ha come matrice di partenza X quella di varianza-covarianza delle variabili. Il procedimento prevede di calcolare gli autovalori di X e di riordinarli in ordine decrescente: il primo autovalore sarà la varianza della prima componente principale, il secondo la varianza della seconda componente principale, e così via, fino ad arrivare all' n -esimo autovalore. Successivamente per ogni autovalore verrà calcolato l'autovettore corrispondente, i cui coefficienti vengono detti *loadings*, e si andrà a costruire la matrice degli autovettori V che ci permetterà di trovare le coordinate di ciascun punto nel nuovo spazio tramite la relazione: $W = V \cdot X$

Le componenti principali ottenute dalla PCA presentano delle proprietà favorevoli, in particolare:

- catturando le direzioni di massima varianza nei dati, permettono di comprimere i dati limitando il più possibile la perdita di informazioni;
- risultano essere scorrelate, il che può facilitarne la futura interpretazione e l'analisi statistica.

Tuttavia questa tecnica presenta delle limitazioni:

- tutti i *loadings* sono tipicamente non nulli, il che potrebbe portare ad una difficile interpretazione del risultato;
- la presenza di dati ad alta dimensionalità potrebbe dar luogo a stime incoerenti dei carichi.

Questi vincoli hanno dato il via alla formulazione di nuove tecniche: Vines¹ ipotizzò che i *loadings* potessero assumere solo i valori 0, 1 e -1. Cadima e

¹Vines, "Simple principal components" (2000)

Jolliffe², invece, una volta calcolate le componenti principali per mezzo della PCA, fissarono a zero tutti i pesi con un valore assoluto minore di un certa soglia. Il problema di questi metodi sta nel fatto che essi costituiscono metodi ad hoc e possono essere fuorvianti specialmente quando si ha a disposizione un numero limitato di campioni. Nacque così l'esigenza di avere a disposizione approcci migliori, che incorporassero il vincolo di sparsità direttamente nella formulazione del problema.

Vennero presentati algoritmi come SCoTLASS e SLRA con la speranza di trovare componenti principali modificate con un elevato numero di *loadings* nulli. Infine Zou, Hastie e Tibshirani proposero un nuovo approccio che prese il nome di *Sparse PCA*, cioè analisi delle componenti principali sparse, abbreviato SPCA.

2.2 Formulazione dell'algoritmo

Introduciamo allora il problema di *Sparse PCA* e vediamo alcuni algoritmi in grado di risolverlo.

Una prima formulazione del modello di PCA consiste nel trovare i vettori $p \in R^n$ e $q \in R^N$ che siano in grado di risolvere il seguente problema di ottimizzazione:

$$(2.1) \quad \min_{p,q} \| M - pq^T \|_F,$$

dove $M \in R^{(n \times N)}$, con n numero di termini e N numero di documenti, è la matrice dei dati (in questo caso il suffisso F indica la norma matriciale di Frobenius). Una volta trovati i vettori p e q , che rappresentano rispettivamente il primo topic e la corrispondente distribuzione nei documenti, la matrice M viene trasformata in una nuova matrice M_1 , alla quale verrà riapplicato l'algoritmo di PCA in modo da ottenere il secondo topic e la corrispondente distribuzione, e così via. La matrice dei dati M è, nella maggior parte dei

²Cadima e Jolliffe, "*Variable selection and the interpretation of principal subspaces*" (2016)

casi, sparsa, a differenza dei vettori p e q che presentano una struttura non sparsa, in quanto sono calcolati per mezzo della PCA standard. Tuttavia sarebbe interessante avere una struttura sparsa anche per questi due vettori in modo da facilitare l'interpretazione del risultato. In particolare si vorrebbe avere un vettore p contenente solo un numero limitato di termini che siano in grado di descrivere un topic, e un vettore q costituito dai soli documenti che trattano quel particolare topic. Il problema di *Sparse PCA*³ viene incontro a queste richieste in quanto, come abbiamo visto, incorpora il vincolo di sparsità direttamente nella formulazione del problema. In questo caso bisogna aggiungere due vincoli di sparsità al problema di minimizzazione mostrato in (2.1), si ottiene perciò il seguente problema di ottimizzazione:

$$(2.2) \quad \min_{p,q} \| M - pq^T \|_F: \quad \| p \|_0 \leq k, \quad \| q \|_0 \leq h$$

La notazione $\|\cdot\|_0$ indica il numero di componenti diverse da 0, k e h , invece, costituiscono i limiti superiori per la cardinalità desiderata.

Shen e Huang (2006) proposero un metodo, denominato "the thresholded power iteration method", in grado di risolvere il problema di minimizzazione (2.2), il cui procedimento consiste nell'alternare su due variabili p e q un'operazione di thresholding ad una di proiezione. In formule:

$$[T_k(Mq)]_\beta \rightarrow p,$$

$$[T_h(M^T p)]_\beta \rightarrow q,$$

dove $T_t(v)$ è l'operatore di thresholding sul vettore v che restituisce un vettore v' della stessa dimensione di v ma con tutte le componenti pari a 0, escluse le t componenti più grandi; $[v]_\beta = \frac{v}{\|v\|_2}$, invece, rappresenta la proiezione di v sulla palla unitaria Euclidea β . Questo metodo dovrebbe essere più veloce del rispettivo usato per la PCA poichè le componenti p e q rimangono sparse ad ogni iterazione.

Zhang e El Ghaoui, invece, presentarono un approccio diretto, denominato

³Zhang et al. (2012), Zou et al. (2006)

DSPCA, in grado di migliorare la sparsità delle componenti principali incorporando questo fattore direttamente nella formulazione del problema di PCA. Il modello da loro proposto presenta la seguente forma:

$$(2.3) \quad \phi = \max_Z \text{Tr} \Sigma Z - \lambda \|Z\|_1 \quad : \quad Z \succeq 0, \text{Tr} Z = 1,$$

dove Σ è la matrice di covarianza (matrice semidefinita positiva di dimensione $n \times n$) e $\lambda \geq 0$ è un parametro che incoraggia la sparsità. Questo problema risulta essere un rilassamento del seguente problema di PCA:

$$(2.4) \quad \psi = \max_x x^T \Sigma x - \lambda \|x\|_0 \quad : \quad \|x\|_2 = 1,$$

dove $\|x\|_0$ indica il numero di elementi diversi da 0 in x . Per dimostrare quanto appena affermato consideriamo il problema equivalente a (2.4), dato da:

$$(2.5) \quad \max_Z \text{Tr} \Sigma Z - \lambda \sqrt{\|Z\|_0} \quad : \quad Z \succeq 0, \text{Tr} Z = 1, \text{Rank}(Z) = 1$$

Le due formulazioni risultano essere analoghe: se Z^* è soluzione di (2.5) allora $Z \succeq 0$ e $\text{Rank}(Z) = 1$ portano a $Z = xx^T$, da cui segue che $\|Z\|_0$ è equivalente a $(\|x\|_0)^2$; inoltre $\text{Tr} Z = 1$ implica $\|x\|_2 = 1$.

Terminiamo la dimostrazione applicando la disuguaglianza

$$\|Z\|_1 \leq \sqrt{\|Z\|_0} \|Z\|_F = \sqrt{\|Z\|_0},$$

che ci consente di riscrivere il problema (2.5) come:

$$(2.6) \quad \max_Z \text{Tr} \Sigma Z - \lambda \|Z\|_1 \quad : \quad Z \succeq 0, \text{Tr} Z = 1, \text{Rank}(Z) = 1$$

La funzione obiettivo di (2.3) non è convessa e perciò il problema diventa molto difficile da risolvere: nel seguito svolgeremo diversi passaggi che ci consentiranno di lavorare con un problema più facile da trattare.

Per prima cosa possiamo sfruttare il seguente teorema:

Data $\Sigma = A^T A$, con $A = (a_1, \dots, a_n) \in R^{m \times n}$, possiamo riscrivere il problema (2.4) come:

$$\psi = \max_{\|\xi\|_2=1} \sum_{i=1}^n ((a_i^T \xi)^2 - \lambda)_+$$

Osserviamo che l' i -esima caratteristica è assente all'ottimo se $(a_i^T \xi)^2 \leq \lambda$ per ogni ξ tale che $\|\xi\|_2 = 1$. Questo teorema ci consente di rimuovere la caratteristica i -esima, con $i \in \{1, \dots, n\}$ se $\Sigma_{ii} = a_i^T a_i < \lambda$ (Σ_{ii} indica la varianza della componente i -esima). Grazie a questo risultato d'ora in poi possiamo assumere, senza perdita di generalità, che

$$\lambda < \sigma_{min}^2 := \min_{1 \leq i \leq n} \Sigma_{ii}$$

La prima trasformazione di (2.3) consiste nel cambio di variabili dato da $X = \gamma Z$, con $\gamma \geq 0$, da cui otteniamo:

$$(2.7) \quad \phi = \max_{\gamma^{-1}X} \frac{1}{\gamma} (Tr \Sigma X - \lambda \|X\|_1) \quad : \quad X \succeq 0, Tr X = \gamma$$

Possiamo a questo punto ottimizzare rispetto $\gamma \geq 0$, ottenendo così $\gamma = \phi$. Sostituiamo questo valore all'espressione precedente e otteniamo:

$$\phi^2 = \max_X Tr \Sigma X - \lambda \|X\|_1 \quad : \quad X \succeq 0, Tr X = \phi,$$

che può essere riscritto come:

$$\phi^2 = \max_X Tr \Sigma X - \lambda \|X\|_1 + \frac{1}{2}\phi^2 - \frac{1}{2}(Tr X)^2 \quad : \quad X \succeq 0$$

Facciamo un altro passaggio e otteniamo:

$$(2.8) \quad \frac{1}{2}\phi^2 = \max_X Tr \Sigma X - \lambda \|X\|_1 - \frac{1}{2}(Tr X)^2 \quad : \quad X \succeq 0$$

Di conseguenza per trovare la soluzione ottima Z^* di (2.3) bisognerà procedere nel seguente modo: si dovrà calcolare la soluzione ottima X^* di (2.8) ed effettuare la trasformazione $Z^* = \frac{X^*}{\phi} = \frac{X^*}{Tr X^*}$.

Non resta che riscrivere il problema (2.8) in modo da lavorare con una funzione obiettivo strettamente convessa:

$$(2.9) \quad \max_X Tr \Sigma X - \lambda \|X\|_1 - \frac{1}{2}(Tr X)^2 + \beta \log \det X \quad : \quad X \succ 0$$

dove $\beta > 0$ è un parametro di penalità. Possiamo ora applicare a quest'ultima formulazione il metodo riga per riga (in inglese "the row-by-row method") con il quale aggiorneremo, una alla volta, ogni coppia riga/colonna di X . A tal proposito scriviamo le matrici X e Σ come:

$$X = \begin{pmatrix} Y & y \\ y^T & x \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} S & s \\ s^T & \sigma \end{pmatrix}$$

dove $Y, S \in R^{(n-1) \times (n-1)}$, $y, s \in R^{n-1}$ e $x, \sigma \in R$, con Y fissa e $(y, x) \in R^n$ variabile. Sostituendo in (2.9) otteniamo:

(2.10)

$$\psi := \max_{x,y} 2(y^T s - \lambda \|y\|_1) + (\sigma - \lambda)x - \frac{1}{2}(TrY + x)^2 + \beta \log(x - y^T Y^{-1} y) : y \in Range(Y)$$

Al fine di semplificare il problema consideriamo la relazione

$$\log \eta + 1 = \min_{z>0} z\eta - \log \eta$$

e poniamo

$$f(x) := (\sigma - \lambda)x - \frac{1}{2}(TrY + x)^2$$

così da ottenere:

(2.11)

$$\psi = \max_{y \in Range(Y)} 2(y^T s - \lambda \|y\|_1) + f(x) + \beta \left(\min_{z>0} z(x - y^T Y^{-1} y) - \log z - 1 \right)$$

A seguire presentiamo qualche calcolo che ci consentirà di risolvere il problema appena ricavato:

$$\psi + \beta = \max_{y \in Range(Y)} 2(y^T s - \lambda \|y\|_1) + f(x) + \beta \min_{z>0} (z(x - y^T Y^{-1} y) - \log z) =$$

$$\min_{z>0} \max_{y \in Range(Y)} 2 \left(y^T s - \lambda \|y\|_1 - \frac{\beta z}{2} y^T Y^{-1} y \right) + \max_x (f(x) + \beta z x) - \beta \log z =$$

$$\min_{z>0} h(z) + 2g(z)$$

Risolviamo separatamente $h(z)$ e $g(z)$, per $z > 0$, e otteniamo:

$$\begin{aligned} h(z) &:= -\beta \log z + \max_x (f(x) + \beta z x) = \\ &= -\beta \log z + \max_x ((\sigma - \lambda + \beta z)x - \frac{1}{2}(TrY + x)^2) = \\ &= -\beta \log z + \max_x ((\sigma - \lambda - TrY + \beta z)x - \frac{1}{2}x^2) \end{aligned}$$

Calcoliamo l'ottimo di

$$\begin{aligned} &\max_x ((\sigma - \lambda - TrY + \beta z)x - \frac{1}{2}x^2) \\ &(\sigma - \lambda - TrY + \beta z) - x = 0 \end{aligned}$$

da cui segue che:

$$(2.12) \quad x^* = \sigma - \lambda - TrY + \beta z$$

Possiamo concludere che $h(z)$ si può scrivere come:

$$h(z) := -\frac{1}{2}(TrY)^2 - \beta \log z + \frac{1}{2}(\sigma - \lambda - TrY + \beta z)^2$$

Svolgiamo ora i calcoli per $g(z)$:

$$\begin{aligned} g(z) &:= \max_{y \in Range(Y)} y^T s - \lambda \|y\|_1 - \frac{\beta z}{2} y^T Y^{-1} y = \\ &= \max_{y \in Range(Y)} y^T s + \min_{v: \|v\|_\infty \leq \lambda} y^T v - \frac{\beta z}{2} y^T Y^{-1} y = \\ &= \min_{v: \|v\|_\infty \leq \lambda} \max_{y \in Range(Y)} (y^T (s + v) - \frac{\beta z}{2} (y^T Y^{-1} y)) = \\ &= \min_{u: \|u-s\|_\infty \leq \lambda} \max_{y \in Range(Y)} (y^T u - \frac{\beta z}{2} (y^T Y^{-1} y)) \end{aligned}$$

L'ottimo di

$$\max_{y \in Range(Y)} y^T u - \frac{\beta z}{2} (y^T Y^{-1} y)$$

si ottiene ponendo $u - \beta z Y^{-1} y$ uguale a 0, avremo perciò:

$$(2.13) \quad y^* = \frac{1}{\beta z} Y u,$$

da cui segue che:

$$g(z) := \min_{u: \|u-s\|_\infty \leq \lambda} \frac{1}{2\beta z} u^T Y u$$

Possiamo concludere riscrivendo la formulazione (2.11) come:

$$\psi + \beta = \min_{u,z} -\frac{1}{2}(TrY)^2 - \beta \log z + \frac{1}{2}(\sigma - \lambda - TrY + \beta z)^2 + \frac{1}{\beta z} u^T Y u \quad : \quad z > 0, \|u-s\|_\infty \leq \lambda$$

Facciamo ancora qualche sostituzione, ponendo $c := \sigma - \lambda - TrY$ otteniamo:

$$\psi + \beta = \min_{u,z} -\frac{1}{2}(TrY)^2 - \beta \log z + \frac{1}{2}(c + \beta z)^2 + \frac{1}{\beta z} u^T Y u \quad : \quad z > 0, \|u-s\|_\infty \leq \lambda$$

Applicando ancora $\psi' := \psi + \beta + \frac{1}{2}(TrY)^2$ riscriviamo il problema precedente come:

$$\psi' = \min_{u,z} -\beta \log z + \frac{1}{2}(c + \beta z)^2 + \frac{1}{\beta z} u^T Y u \quad : \quad z > 0, \|u-s\|_\infty \leq \lambda$$

Infine effettuiamo un'ultima sostituzione, $\tau = \beta z$, e ricaviamo la formulazione finale del problema di ottimizzazione:

(2.14)

$$\psi' - \beta \log \beta = \min_{u,\tau} -\beta \log \tau + \frac{1}{2}(c + \tau)^2 + \frac{1}{\tau} u^T Y u \quad : \quad \tau > 0, \|u-s\|_\infty \leq \lambda$$

Per facilitare la risoluzione iniziamo a prendere in considerazione $R^2 := \min_u u^T Y u \quad : \quad \|u-s\|_\infty \leq \lambda$. Dati $\hat{Y} \in R^{(n-2) \times (n-2)}$, $\hat{x}, \hat{y}, \hat{s} \in R^{n-2}$, $y_1, s_1 \in R$ valori fissati e $\eta \in R$ variabile, consideriamo le seguenti partizioni:

$$u = \begin{pmatrix} \eta \\ \hat{x} \end{pmatrix},$$

$$Y = \begin{pmatrix} y_1 & \hat{y}^T \\ \hat{y} & \hat{Y} \end{pmatrix},$$

$$s = \begin{pmatrix} s_1 \\ \hat{s} \end{pmatrix}$$

e applichiamo un algoritmo di discesa delle coordinate che aggiorni la prima coordinata di u . Così facendo R^2 diventa:

$$\min_{\eta} y_1 \eta^2 + (2\hat{y}^T \hat{u}) \eta \quad : \quad \|\eta - s_1\|_\infty \leq \lambda$$

Questo problema può essere risolto analiticamente per η usando la formula:

$$\begin{cases} -\frac{\hat{y}^T \hat{u}}{y_1} & \text{se } \|s_1 + \frac{\hat{y}^T \hat{u}}{y_1}\| \leq \lambda \\ s_1 - \lambda & \text{se } -\frac{\hat{y}^T \hat{u}}{y_1} < s_1 - \lambda, y_1 > 0 \text{ o se } \hat{y}^T \hat{u} > 0, y_1 = 0 \\ s_1 + \lambda & \text{se } -\frac{\hat{y}^T \hat{u}}{y_1} > s_1 + \lambda, y_1 > 0 \text{ o se } \hat{y}^T \hat{u} \leq 0, y_1 = 0 \end{cases}$$

Possiamo allora risolvere il problema finale dato da:

$$\min_{\tau > 0} \frac{R^2}{\tau} - \beta \log \tau + \frac{1}{2}(c + \tau)^2$$

Una volta trovato τ^* possiamo ricavare le variabili x e y nel seguente modo: dalla formula (2.12) e ricordando che $\tau = \beta z$ otteniamo $y = \frac{1}{\tau} Y u$, mentre dalla (2.13) abbiamo che $x = \sigma - \lambda - Tr Y + \tau$.

Capitolo 3

Metodologia

In questa sezione verrà presentata la metodologia di *topic modeling* proposta affinché i documenti presenti in una collezione di dati testuali vengano raggruppati automaticamente in gruppi coerenti e ben separati. Analizzeremo pertanto come il lavoro di esperti ed analisti, che consiste nella configurazione e nell'inserimento di parametri specifici per ogni algoritmo, verrà sostituito da tecniche automatiche. In questo modo il processo di *mining* diventa efficiente, non si avrà più il bisogno di una continua visione umana dell'attività. Le fasi principali della metodologia di *topic modeling* proposta sono schematizzate nella figura 3.1 e verranno trattate nello specifico nei paragrafi che seguono.

3.1 *Pre-processing*

Per prima cosa bisogna fare in modo che i dati testuali abbiano una struttura tale da poter applicare gli algoritmi di *data-mining*: sarà quindi necessario effettuare un processo di trasformazione che prende il nome di *pre-processing* (pre-elaborazione). Questa operazione consiste in più fasi che devono essere eseguite con un certo ordine in base al tipo di documenti da analizzare:

- **Suddivisione del documento:** i documenti possono essere suddivisi in frasi, paragrafi, oppure essere analizzati a partire da tutto il loro

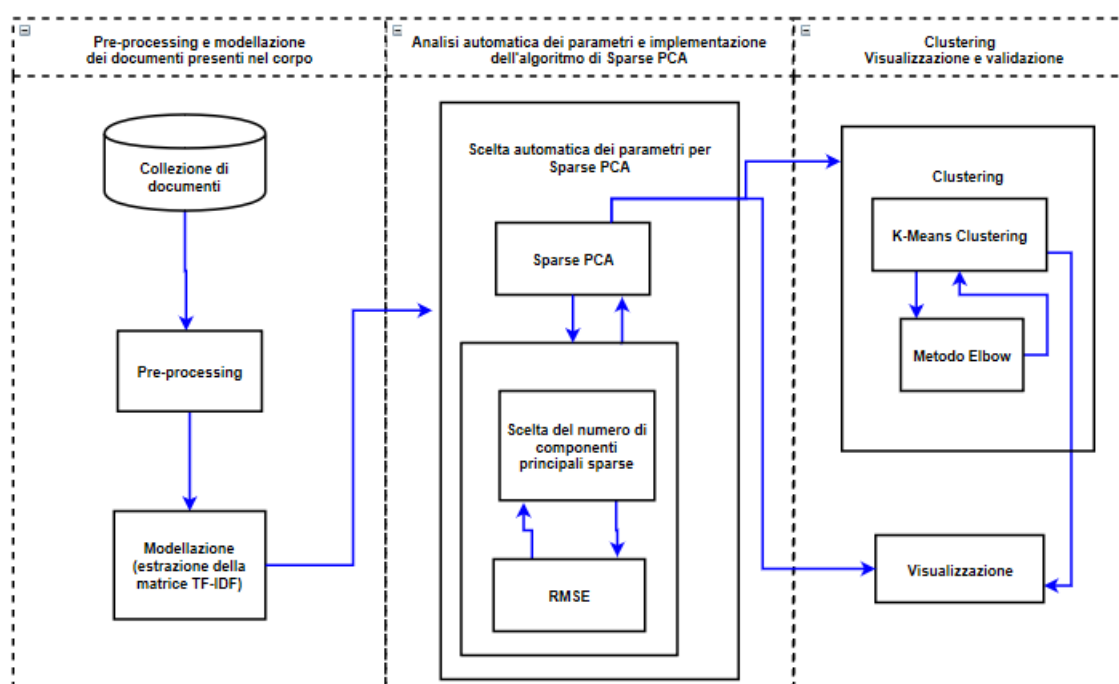


Figura 3.1: Fasi principali della metodologia di *topic modeling*

contenuto.

- Tokenizzazione:** ogni documento viene segmentato in unità linguistiche (parole, punteggiatura, numeri, etc.) che prendono il nome di '*token*'. In questa fase bisogna prestare particolare attenzione nel definire i delimitatori che separano ogni *token*: una scelta sbagliata potrebbe scaturire la presenza di errori, il che può causare problemi ai fini dell'analisi. Ad esempio se si sta analizzando un documento scritto in una lingua come l'italiano o l'inglese si potrà usare il carattere di spazio come possibile delimitatore, sarà invece necessario individuarne un altro nel caso in cui il documento fosse scritto, ad esempio, in cinese o in giapponese. Inoltre potrebbe accadere che il solo carattere di spazio non sia sufficiente e che ad esso debba essere aggiunto, ad esempio, un carattere di punteggiatura o una parentesi. Di conseguenza bisogne-

rebbe creare degli strumenti di tokenizzazione specifici, in base al tipo di documento preso in considerazione.

- **Rimozione delle *stop-words*:** una volta ricavati i *tokens* sarà necessario rimuovere tutti quei termini che non sono rilevanti ai fini dell'analisi in quanto non forniscono nessun tipo di informazione aggiuntiva. In questa lista troviamo numeri, articoli, avverbi, preposizioni, congiunzioni, caratteri speciali e tutte quelle parole che risultano essere comuni (per esempio '*are*', '*can*', '*have*', '*do*', etc.).
- ***Stemming*:** in questa fase ogni parola viene ridotta alla forma base, o alla radice, per mezzo della rimozione dei prefissi, dei suffissi e della desinenza. Questo procedimento dipende dal linguaggio usato nel documento e consente di diminuire il numero di termini distinti della collezione e di aumentare il conteggio delle occorrenze di certe parole. Tuttavia bisogna fare attenzione a non incorrere ad un '*over-stemming*' o ad un '*under-stemming*'. Il primo termine indica il processo di *stemming* caratterizzato da un incremento del grado di *stem*, cioè da un accorciamento della radice del termine: si intensificano i legami tra le parole con lo stesso significato ma si rischia di includere anche quelle con significato diverso, arrivando così alla perdita della specificità. D'altra parte l'*under-stemming* è caratterizzato da una riduzione del grado di *stem*, cioè da un allungamento della radice: in questo caso i termini risultano essere più specifici ma si rischia di perdere qualche relazione tra le parole che hanno lo stesso significato, arrivando ad una perdita di generalità.
- **Normalizzazione:** infine i termini vengono convertiti alla forma minuscola (o maiuscola).

3.2 Modellazione del documento

In questo paragrafo andremo a definire diverse funzioni di peso che ci saranno utili in fase di analisi della struttura di una collezione di documenti testuali. Iniziamo col fornire una serie di notazioni: sia $D = \{W_1, W_2, \dots, W_M\}$ una collezione di M documenti, detta corpo, dove $W_i = (w_1, w_2, \dots, w_N)$ indica un generico documento caratterizzato da N termini w_j , e sia $V = \{t_1, t_2, \dots, t_{|V|}\}$ l'insieme dei termini distinti all'interno della collezione. Possiamo allora rappresentare il corpo come una matrice che chiameremo "frequenza-termine", indicata con FT , le cui righe corrispondono ai documenti W_i , con $i = 1, 2, \dots, N$, e le colonne ai singoli termini t_i , con $i = 1, 2, \dots, |V|$, e la cui entrata $x_{i,j}$ indica il numero di occorrenze del termine j -esimo nel documento i -esimo. Definiamo a questo punto due vettori che chiameremo "frequenza-documento" e "frequenza-collezione", indicati rispettivamente con FD e FC . Entrambi hanno un numero di componenti pari al numero di termini differenti nella collezione: il primo indica il numero di documenti in cui il termine t_i appare, mentre il secondo il numero totale di occorrenze del termine nel corpo. In particolare la matrice FT indica l'importanza di ogni termine in un documento, mentre il vettore FD può essere visto come un indicatore di assenza di informazione di un termine. Infine prendiamo in considerazione la matrice "TF-IDF" in cui ogni componente $(TF - IDF)_{i,j}$ assume valore

$$(3.1) \quad \begin{cases} (1 + \log((FT)_{i,j})) \log \frac{M}{(FD)_i} & \text{se } (FT)_{i,j} \leq 1 \\ 0 & \text{altrimenti} \end{cases}$$

Questa funzione di peso aumenta proporzionalmente al numero di volte in cui il termine è contenuto nel documento, ma diminuisce in maniera inversamente proporzionale alla frequenza del termine nella collezione. Inoltre, in alcuni casi, le colonne vengono divise per la loro norma Euclidea al fine di normalizzarle.

3.3 *Sparse PCA*

A questo punto non resta che applicare l'algoritmo di *Sparse PCA* in modo da individuare documenti che appartengono allo stesso topic e per ogni topic una lista di termini che meglio lo descrivono. Il modello consente di trovare direzioni sparse che spiegano un'alta varianza: non è altro che una variante della PCA. Si andrà così a definire un numero di componenti principali sparse che ci consentiranno di lavorare su un dataset di dimensione ridotta. In particolare una volta scelti il numero di topic K e il parametro di sparsità α il modello di *Sparse PCA* verrà calcolato a partire dalla matrice "TF-IDF" prima calcolata. Terminata l'implementazione si potranno estrarre due matrici principali: la matrice dei caricamenti (*loadings*) p e quella dei risultati (*scores*) q . La prima ha per righe il numero di componenti principali sparse (cioè il numero K di topic individuati) e per colonne il numero totale di termini distinti presenti nella collezione: l'entrata $p_{i,j}$ indica il peso con cui il termine j -esimo influenza la componente i -esima. Avendo a che fare con componenti sparse molte entrate di p saranno pari a 0, ovviamente maggiore è il valore del parametro α , maggiore sarà il numero di entrate nulle e maggiore sarà pertanto l'interpretabilità del risultato. Possiamo allora riordinare in ordine decrescente ogni vettore $p_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,|V|}\}$ rispetto al valore assoluto di ogni entrata in modo da ottenere per ogni topic i una lista di termini che meglio lo caratterizzano. La matrice q , invece, ha per righe il numero di topic e per colonne il numero di documenti caratterizzanti il corpo: in questo caso per ogni documento j otterremo un vettore q_j le cui componenti rappresentano le sue coordinate rispetto alle K componenti principali. Quest'ultima matrice ci consentirà di rappresentare i documenti in un sistema di riferimento basato sulle coordinate: ognuna di esse individuerà la posizione di uno specifico documento nello spazio. Consideriamo ad esempio il caso in cui il numero di topic K sia pari a 2, di conseguenza la matrice q rappresenterà ogni documento come una coppia di coordinate $(q_{1,j}, q_{2,j})$ che potranno essere visualizzate in un sistema di riferimento bidimensionale. In questo modo si formeranno gruppi di documenti ben coesi e ben separati tra

loro ognuno dei quali farà riferimento ad un topic ben specifico: saremo così in grado di individuare documenti che appartengono allo stesso topic.

3.4 Scelta dei parametri

La scelta dei parametri K e α è un processo che richiede particolare attenzione in quanto bisogna trovare il giusto trade-off tra interpretabilità dei risultati e percentuale di varianza spiegata. In questa sezione verrà presentato il procedimento da noi usato per facilitare la scelta di questi valori. Verrà proposta una nuova formula per il calcolo della varianza totale spiegata dalla matrice q , che prende il nome di varianza aggiustata.

Nel processo di PCA ordinario le componenti principali sono scorrelate tra loro e i caricamenti sono ortogonali, perciò la varianza totale spiegata da q sarà data da $Tr(q^T q)$. Questo procedimento è sconsigliato nel momento in cui si ha a che fare con componenti sparse correlate tra loro: la varianza totale calcolata costituirebbe un risultato ottimistico. Indichiamo con q_i , dove $i = 1, 2, \dots, K + 1$, le prime $K + 1$ componenti principali modificate e supponiamo di voler calcolare la varianza totale spiegata da esse. Supponiamo inoltre che la componente q_{K+1} sia correlata con una delle q_i : in questo caso la varianza della componente q_{K+1} contiene contributi già forniti dalle q_i , i quali non dovrebbero essere inclusi nel calcolo della varianza totale. Presentiamo pertanto una nuova formula che tenga conto della correlazione in q . Indichiamo con $q_{j \cdot 1, \dots, j-1}$ il residuo di q_j , dopo aver aggiustato gli effetti di $q_1, q_2, \dots, q_{j-1}$, definito come:

$$q_{j \cdot 1, \dots, j-1} = q_j - H_{1, \dots, j-1} q_j,$$

dove $H_{1, \dots, j-1}$ è la matrice di proiezione su $q_i, i = 1, 2, \dots, j - 1$. La varianza aggiustata di q_j sarà data da $\|q_{j \cdot 1, \dots, j-1}\|^2$ e la varianza totale spiegata verrà definita come $\sum_{j=1}^K \|q_{j \cdot 1, \dots, j-1}\|^2$. La varianza aggiustata può essere facilmente calcolata facendo riferimento alla decomposizione QR della matrice q , con

Q matrice ortonormale e R triangolare superiore. Vale la seguente relazione:

$$\|q_{j \cdot 1, \dots, j-1}\|^2 = R_{jj}^2,$$

da cui segue che la varianza aggiustata totale spiegata è pari a $\sum_{j=1}^K R_{jj}^2$.

Nel caso in cui le componenti principali di q non siano correlate tra loro la formula della varianza rimarrebbe $Tr(q^T q)$.

Possiamo pertanto presentare la strategia adoperata per la scelta dei parametri. Per prima cosa abbiamo implementato l'algoritmo di *Sparse PCA* per diversi valori di α e, per ognuno di essi, abbiamo calcolato la varianza aggiustata facendo variare il valore assunto da K . Successivamente abbiamo riportato i risultati ottenuti in un grafico analogo a quello presentato in figura 3.2.

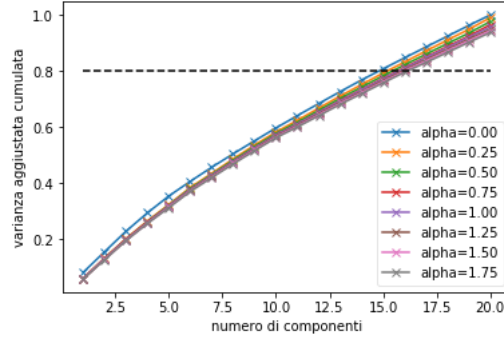


Figura 3.2: Scelta dei parametri K e α

Come si può osservare per ogni valore di α abbiamo una curva; in particolare aumentando il parametro α , cioè la sparsità delle componenti, avremo curve caratterizzate da una minore varianza aggiustata spiegata. Da questo grafico possiamo effettuare la scelta del parametro K , cioè del numero di componenti principali sparse, in modo da spiegare una certa percentuale di varianza.

Per scelta del parametro α , invece, ci viene in aiuto l'RMSE (errore quadratico medio, in inglese *Root Mean Square Error*), una misura di errore assoluta in cui le deviazioni vengono elevate al quadrato in modo da evitare che valori positivi e negativi possano annullarsi a vicenda. Formalmente è definito nel

seguinte modo:

$$RMSE = \sqrt{\sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}},$$

dove $\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n$ indicano i valori predetti, $y_1, y_2, \dots, y_n$ i valori osservati e n è il numero di osservazioni. Questa misura ha il beneficio di penalizzare errori di grossa dimensione in quanto essi vengono amplificati.

Nel nostro caso calcoleremo l'RMSE facendo riferimento alla distanza di ogni curva da quella con $\alpha = 0$. Quest'ultima coincide con il risultato che si otterrebbe dall'algoritmo di PCA standard, in quanto ponendo $\alpha = 0$ non viene preso in considerazione il fattore sparsità. Pertanto per ogni α verrà calcolato l'RMSE e poi i risultati verranno presentati in un grafico analogo a quello mostrato in figura 3.3.

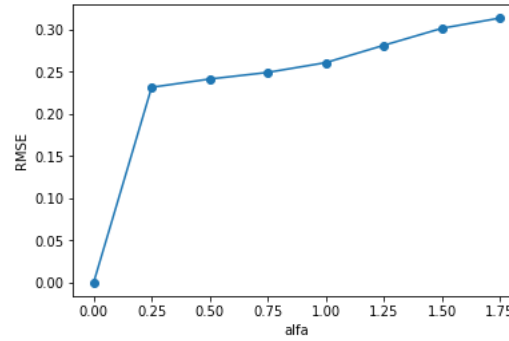


Figura 3.3: Scelta del parametro α

Di conseguenza sceglieremo quel valore di α che non provochi un cambiamento sostanziale nell'inclinazione della curva; nell'esempio mostrato in figura potremmo optare per un $\alpha = 1$.

Per la scelta di α avremmo potuto procedere calcolando altri indici, come ad esempio il MAE oppure il MAPE. Il primo è l'errore assoluto medio (*Mean Absolute Error*) definito come:

$$MAE = \sum_{i=1}^n \frac{|\hat{y}_i - y_i|}{n}.$$

Quest'indice, a differenza dell'RMSE, dà la stessa importanza ad ogni errore. Il secondo, invece, è l'errore assoluto percentuale medio (*Mean absolute percentage error*) calcolato come:

$$MAPE = \left(\frac{1}{n} \sum_{i=1}^n \frac{|\hat{y}_i - y_i|}{|y_i|} \right) * 100.$$

Dal momento che ogni errore viene diviso per ogni osservazione si può affermare che anche in questo caso gli errori di grossa dimensione avranno un impatto maggiore sull'indice.

3.5 Clustering

In questa sezione, invece, presentiamo l'algoritmo di *K-Means Clustering* che ci consentirà di confrontare i risultati ottenuti dall'algoritmo di *Sparse PCA* con quelli di ESCAPE.

Dalla teoria sappiamo che il clustering è quel processo che consiste nel dividere l'intero dataset in gruppi, chiamati *clusters*, in base alle proprietà di questi dati. È un problema di tipo *unsupervised learning* che presenta le seguenti proprietà:

- tutti i punti all'interno di un cluster devono essere simili gli uni con gli altri;
- i punti situati in clusters differenti, invece, devono essere il più possibile diversi tra loro in modo da avere gruppi più significativi.

La qualità del clustering può essere calcolata per mezzo di metriche di valutazioni. Nel seguito ne presentiamo due: l'inerzia e l'indice di Dunn.

Il primo indice ci dà informazioni sulla distanza dei dati all'interno di un cluster (distanza intra-cluster): per ogni punto all'interno di un cluster viene calcolata la distanza rispetto al centroide del cluster stesso e poi tutte queste distanze vengono sommate. Calcoliamo questo valore per ogni gruppo fino ad ottenere il valore finale di inerzia, dato dalla somma di tutti questi valori.

Ovviamente, minore sarà il valore di inerzia, migliori saranno i nostri cluster. Questo indice fa riferimento alla prima proprietà: cerca di rendere i cluster più compatti, più coesi.

D'altra parte l'indice di Dunn fa in modo che venga soddisfatta la seconda proprietà in quanto viene presa in considerazione la distanza tra i centroidi dei diversi cluster (distanza inter-cluster). Presenta la seguente formulazione:

$$Indice \text{ } Dunn = \frac{\min(distanza \text{ } inter - cluster)}{\max(distanza \text{ } intra - cluster)}$$

Sarà di nostro interesse massimizzare questo indice: maggiore sarà questo valore, migliore sarà la qualità dei cluster.

Possiamo allora presentare il modello di *K-Means Clustering*:

1. per prima cosa bisogna specificare il numero di clusters K : questo passaggio può essere facilitato implementando alcuni metodi come ad esempio il metodo Elbow;
2. successivamente l'algoritmo seleziona, in maniera casuale, il centroide da ogni cluster e assegna ogni punto al centroide più vicino;
3. una volta terminato il passaggio definito al punto 2, ricalcola i centroidi per i nuovi clusters appena calcolati;
4. i punti 2 e 3 vengono ripetuti finchè non viene soddisfatto uno dei seguenti criteri di arresto: i centroidi dei nuovi clusters non cambiano, i punti rimangono nello stesso cluster, viene raggiunto il massimo numero di iterazioni.

L'idea del metodo Elbow è quella di implementare il *K-Means Clustering* sul dataset di riferimento, facendo variare il parametro K in un range di valori e, per ognuno di essi, calcolare l'indice SSE (*Sum of Squared Errors*). Ricordiamo che l'SSE è la somma dei quadrati delle differenze tra ogni osservazione rispetto alla media delle osservazioni nel gruppo, in formule:

$$SSE = \sum_{i=1}^n (x_i - \bar{x})^2$$

Successivamente raffiguriamo i risultati in un grafico analogo a quello mostrato in figura 3.4. Come si può notare la curva ha la forma di un braccio e, di conseguenza, il "gomito" del braccio (*elbow*, da cui deriva il nome del metodo) costituisce il valore di K ideale; questo perchè noi vorremmo avere un valore basso per l'SSE che ci consenta di avere un K tendenzialmente basso in modo da creare gruppi di cluster costituiti da più punti. Tuttavia non sempre questo metodo lavora bene, spesso abbiamo a che fare con curve in cui non è ben chiaro quale sia il miglior valore di K da scegliere.

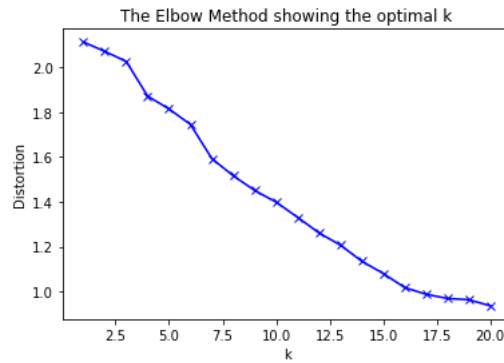


Figura 3.4: Metodo Elbow

3.6 Visualizzazione

Ai fini dell'analisi è importante verificare se un modello ha senso a livello pratico. Questo controllo di qualità può essere effettuato per mezzo della rappresentazione t-SNE (*t-Distributed Stochastic Neighbor Embedding*) e della rappresentazione a nube di parole, *word cloud*.

3.6.1 Rappresentazione t-SNE

La rappresentazione t-SNE (*t-Distributed Stochastic Neighbor Embedding*) è una tecnica non lineare che consente di ridurre la dimensionalità, adatta per la visualizzazione di dataset corposi. L'algoritmo modella i punti in modo

che oggetti vicini nello spazio originale risultino vicini nello spazio a dimensionalità ridotta, e oggetti lontani risultino lontani, cercando di preservare la struttura locale. Per vedere come funziona l'algoritmo prendiamo in considerazione un campione di oggetti $x_1, x_2, \dots, x_N$ di taglia N in uno spazio ad alta dimensionalità. Per prima cosa viene costruita una distribuzione di probabilità p_{ij} che ad ogni coppia di punti nello spazio originale, cioè quello ad alta dimensionalità, associa un valore di probabilità elevato se i due punti sono simili, basso se sono dissimili. La distribuzione p_{ij} è simmetrica nelle due variabili ed è definita come:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N},$$

dove

$$p_{j|i} = \frac{e^{-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}}}$$

L'algoritmo di t-SNE cerca di costruire una mappa d -dimensionale $y_1, y_2, \dots, y_N$ tale per cui i punti riflettano il meglio possibile la similarità p_{ij} nello spazio di partenza. Viene pertanto definita una seconda distribuzione di probabilità analoga nello spazio a dimensione ridotta che indichiamo con q_{ij} . La similarità tra due punti y_i e y_j nello spazio a dimensionalità ridotta viene definita come:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq m} (1 + \|y_k - y_m\|^2)^{-1}}$$

Come si può osservare nello spazio a dimensionalità ridotta si fa riferimento a una distribuzione t di Student con un grado di libertà piuttosto che alla Gaussiana, usata nello spazio ad alta dimensionalità, le cui code pesanti consentono di modellare meglio la dissimilarità tra oggetti distanti. Infine l'algoritmo minimizza la divergenza di Kullback-Leibler delle due distribuzioni tramite un metodo di discesa del gradiente:

$$KL = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Si avranno penalità se punti vicini nello spazio originale, cioè p_{ij} elevato, vengono considerati lontani nello spazio a dimensionalità ridotta, q_{ij} basso.

Il risultato è una mappa a bassa dimensionalità che riflette le similarità tra i punti nello spazio ad alta dimensionalità. Un esempio di visualizzazione del dataset per mezzo della rappresentazione t-SNE è fornito nella figura 3.5: come si può osservare i punti, che in questo caso indicano i documenti presenti in una collezione, vengono raggruppati in cluster diversi, in base al topic che li caratterizza.

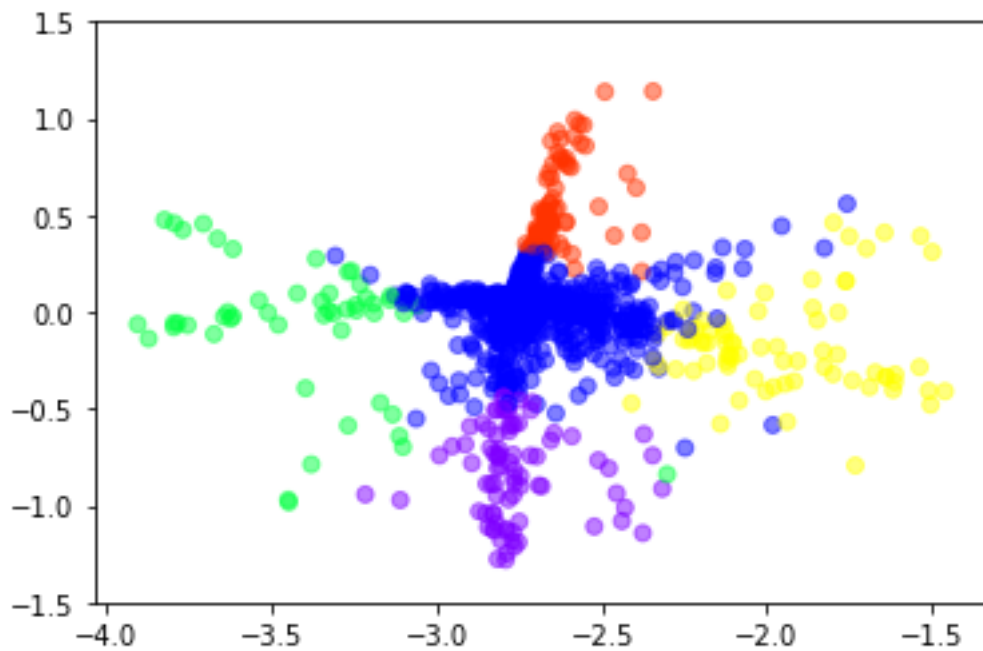


Figura 3.5: Rappresentazione t-SNE

Tuttavia le rappresentazioni t-SNE possono essere fuorvianti: diventa molto importante la scelta dei parametri, tra i più importanti ricordiamo il parametro perplessità (*perplexity*) e il numero di iterazioni. All'aumentare della perplessità otterremo configurazioni più stabili, tuttavia il valore di questo parametro dovrebbe sempre essere minore del numero totale di punti in modo da evitare che l'algoritmo fallisca e che abbia un comportamento inaspettato. Anche la scelta del numero di iterazioni è molto importante in quanto, per interpretare al meglio i risultati, vogliamo che l'algoritmo converga.

3.6.2 Rappresentazione nube di parole

Un altro modello di visualizzazione pratico è la rappresentazione a nube di parole, *word cloud*, nella quale per ogni topic vengono raffigurati i termini principali. Come abbiamo visto, l'algoritmo di *Sparse PCA* ci consente di ottenere per ogni argomento i la lista di parole che lo caratterizzano, ordinata in base ai pesi delle entrate $p_{i,j}$ con $j = 1, 2, \dots, |V|$. I termini in testa alla lista verranno rappresentati con un formato più grande, che diminuirà per gli elementi situati sulla coda: questo tipo di rappresentazione ha un forte impatto visivo che facilita l'individuazione di partizioni buone e non.

Un esempio di visualizzazione di questo tipo è mostrato in figura 3.6: come si può osservare la *word-cloud* illustrata a sinistra, caratterizzata da termini come 'graph', 'finit', 'theorem', 'vector', 'matrix', 'algebra', farà riferimento all'argomento 'Matematica'; sulla destra, invece, termini come 'music', 'chord', 'bass', 'tempo', 'instrument', 'melodi' faranno allusione all'argomento 'Musica'.

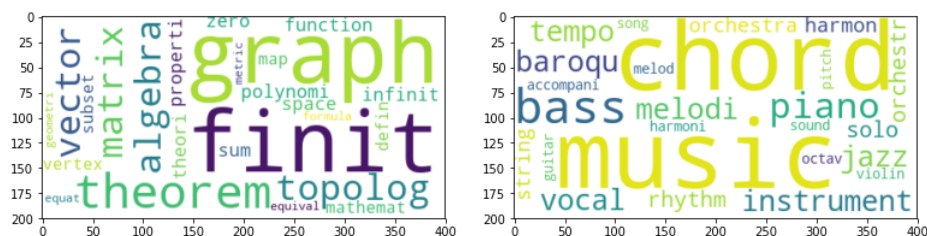


Figura 3.6: Rappresentazione *word-cloud*

Capitolo 4

Risultati

In questo capitolo analizzeremo le performance del modello su tre raccolte di dati reali, con caratteristiche e proprietà testuali diverse: due collezioni di dati testuali raccolti da Wikipedia e una di Tweet. L'algoritmo è stato sviluppato in Python seguendo il procedimento presentato nel capitolo precedente. La prima fase, quella di *pre-processing*, presenta operazioni diverse a seconda del tipo di dataset considerato: ad esempio, sarà necessario rimuovere gli URLs, gli username e il simbolo "#" che precede gli hashtag nel caso di una collezione di Tweet; oppure i paragrafi intitolati *See Also* e *References* nel caso di file Wikipedia, inefficaci ai fini dell'analisi. Successivamente, una volta ottenuta una lista di *tokens* per ogni documento, si potrà procedere con il calcolo della matrice frequenza termine FT per poi passare a quello della matrice $TF-IDF$, ottenuta per mezzo della formula (3.1). Le restanti operazioni, cioè quelle relative alla scelta dei parametri, all'implementazione e alle rappresentazioni dell'algoritmo di *Sparse PCA*, verranno analizzate separatamente nei paragrafi che seguono in base al tipo di collezione presa in considerazione.

4.1 Collezioni Wikipedia

Come primo caso di studio andremo a considerare due collezioni di documenti testuali importati da Wikipedia. Per prima cosa bisognerà scegliere i parametri K e α in modo da implementare l'algoritmo di *Sparse PCA*; a tal proposito facciamo riferimento ai grafici mostrati in figura 4.1, per la prima collezione di dati, e in 4.2, per la seconda.

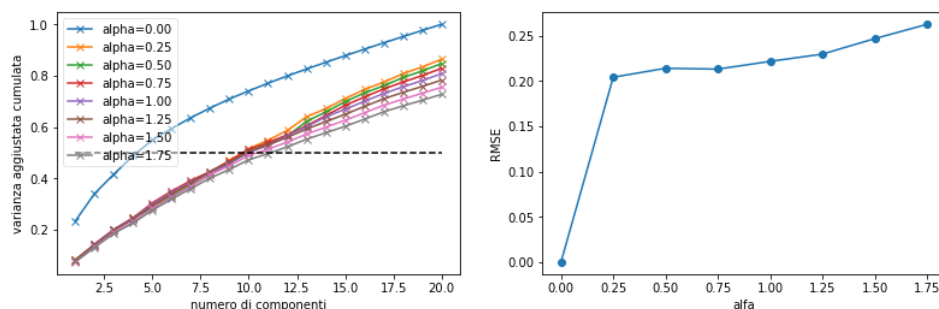


Figura 4.1: Scelta dei parametri K e α per il primo corpo estratto da Wikipedia.

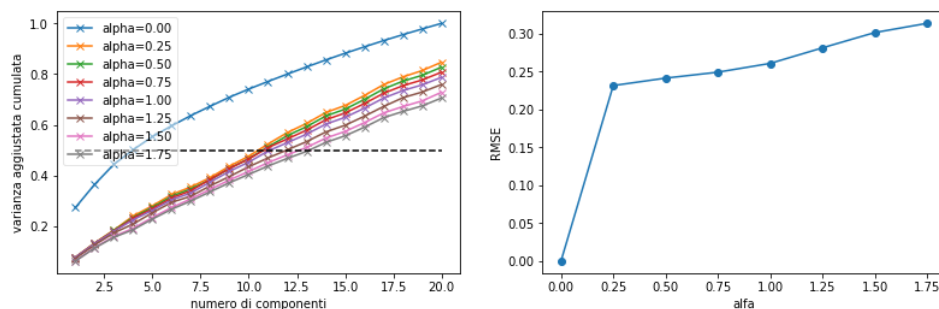


Figura 4.2: Scelta dei parametri K e α per il secondo corpo estratto da Wikipedia

Per quanto riguarda la scelta del K , cioè del numero di componenti principali sparse, richiamiamo il grafico mostrato sulla sinistra e selezioniamo quel valore che ci consenta di spiegare almeno il 50% della varianza totale aggiustata:

per la prima collezione scegliamo un valore di K pari a 10, mentre per la seconda un valore pari a 11. Il grafico mostrato sulla destra, invece, ci consente di selezionare il parametro di sparsità: questa volta sceglieremo quel valore di α tale per cui la curva di RMSE non presenti un'inclinazione notevole. Di conseguenza fisseremo il parametro di sparsità a 0.75 per la prima collezione e a 1 per la seconda. Una volta terminata l'analisi automatica dei parametri proseguiamo con il prossimo step: l'implementazione dell'algoritmo di *Sparse PCA*.

Focalizziamo la nostra analisi sulla prima collezione di dati testuali raccolti da Wikipedia e implementiamo l'algoritmo fissando $K = 10$ e $\alpha = 0.75$; otteniamo così la matrice p dei caricamenti (*loadings*) e quella dei risultati (*scores*), indicata con q . La prima matrice ci consente di rappresentare ogni topic in base ai termini che lo costituiscono, mentre la seconda ci fornisce una raffigurazione dei documenti presenti nel corpo a seconda dei topic che li descrivono. Di conseguenza applichiamo l'algoritmo di *K-Means Clustering* alla matrice degli *scores*, in modo da individuare per ogni documento il cluster (il topic) di appartenenza. Per prima cosa calcoliamo il numero di cluster da prendere in considerazione attraverso il metodo Elbow: facciamo variare k da 1 a 20 e, per ogni valore, implementiamo l'algoritmo di *K-Means Clustering*, calcoliamo l'indice SSE e infine riportiamo i risultati ottenuti nel grafico mostrato in figura 4.3.

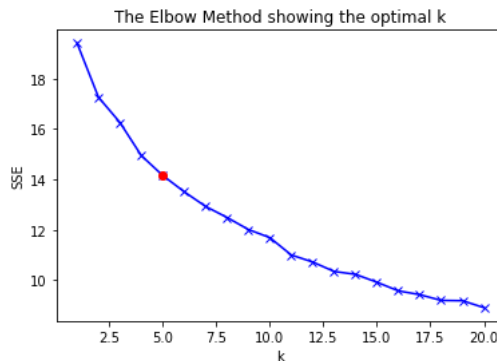


Figura 4.3: Metodo Elbow per la scelta del numero di cluster

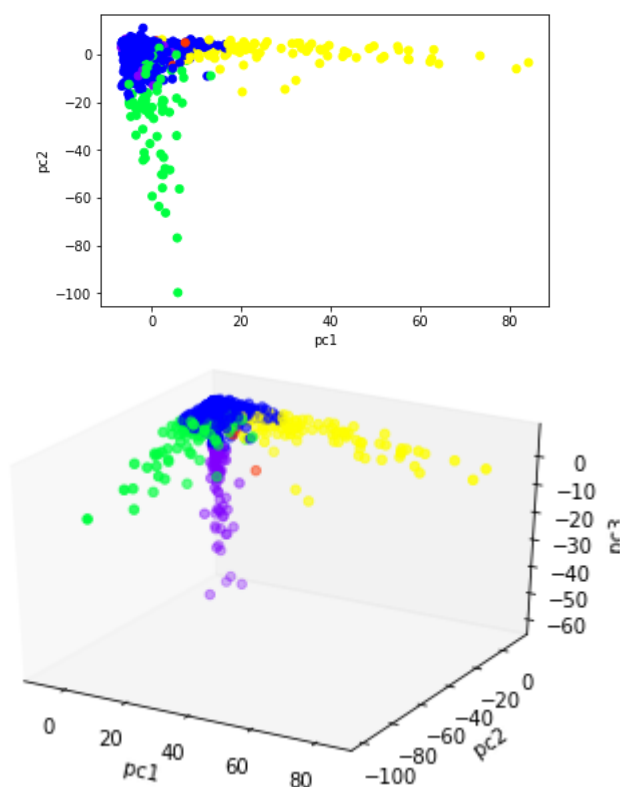


Figura 4.4: Rappresentazione dei documenti in un grafico bidimensionale ed in uno tridimensionale

Osservando il grafico si potrebbe pensare di scegliere un numero di cluster pari a 5 poichè in prossimità di questo valore si ha il "gomito" del "braccio". Di conseguenza implementiamo l'algoritmo di *K-Means Clustering* che ci consentirà di assegnare ad ogni documento un colore che dipenderà dai valori presenti nella matrice q . Possiamo allora fornire una prima rappresentazione grafica in cui ogni documento presente nel corpo viene raffigurato, sotto forma di punto, in un grafico bidimensionale che presenta in ascisse la prima componente principale sparsa, cioè quella che spiega una maggior quantità di varianza aggiustata, e in ordinate la seconda componente principale sparsa. Il grafico è mostrato in figura 4.4: per ogni documento si estrae dalla matrice q il valore corrispondente alla prima componente principale e quello corrispondente alla seconda, che costituiscono rispettivamente la coordinata

delle ascisse e quella delle ordinate; inoltre ogni documento viene colorato in base al cluster a cui appartiene. Sempre nella stessa figura abbiamo fornito una rappresentazione tridimensionale, in questo caso la terza dimensione fa riferimento alla terza componente principale.

Un altro tipo di rappresentazione adatta per la visualizzazione di dataset corposi è la rappresentazione t-SNE, mostrata in figura 4.5, calcolata con un numero di componenti pari a 2, una *perplexity* pari a 900 e un numero di iterazioni uguale a 5000. Questa raffigurazione ci consente di individuare cinque topic e, per ognuno di essi, i documenti che li costituiscono colorati in base al topic di appartenenza. L'algoritmo consente così di raggruppare automaticamente i documenti presenti nella prima collezione in gruppi coerenti e ben separati.

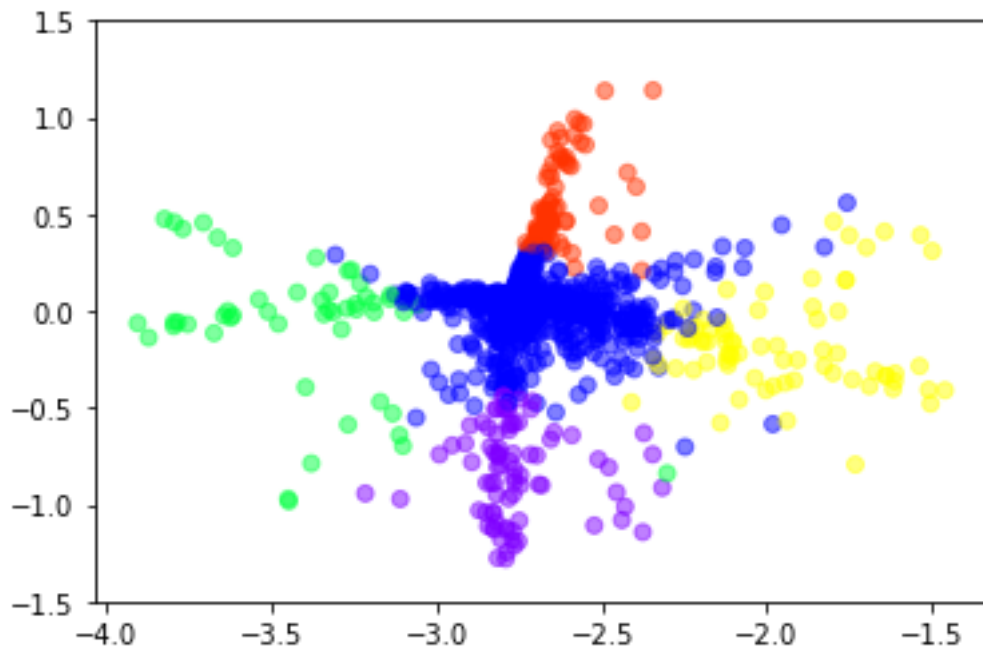


Figura 4.5: Rappresentazione t-SNE

Consideriamo ancora un ultimo tipo di rappresentazione, questa volta ottenuta a partire dalla matrice dei *loadings*. Come abbiamo visto nel capitolo precedente, da questa matrice possiamo ricavare per ogni topic la lista del-

le parole che meglio lo descrivono e rappresentare i risultati ottenuti in un grafico che prende il nome di rappresentazione a nube di parole (*word cloud representation*). Ognuno di questi grafici rappresenta un topic, all'interno del quale vengono raffigurati i termini che lo caratterizzano con un formato che dipende dalla loro posizione nella lista: i termini in testa presentano un formato più grande, perché sono quelli più significativi, che diminuisce man mano che ci avviciniamo alla coda della lista. Questa rappresentazione, riportata nella figura 4.6, ci consente di individuare i topic principali trattati nella prima collezione di dati testuali raccolti da Wikipedia: sport, letteratura, matematica, cucina e musica.

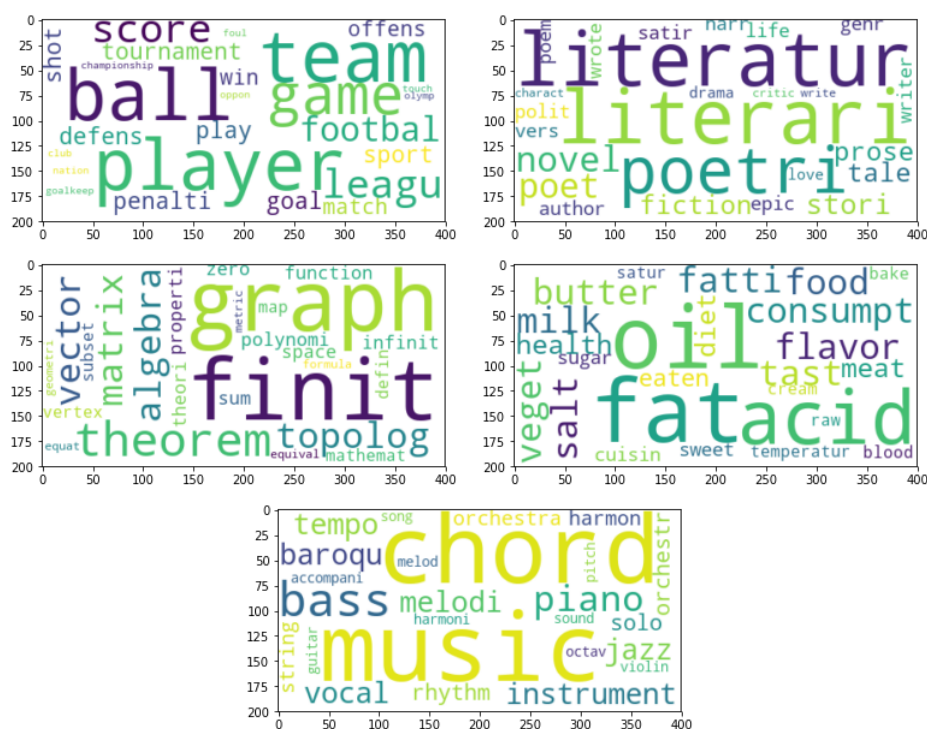


Figura 4.6: Rappresentazione *word-cloud*

L'analisi della seconda collezione di dati testuali raccolti da Wikipedia seguirà lo stesso procedimento visto fin ora. In particolare, una volta ricavati i parametri inerenti il numero di componenti principali K e la sparsità α (fi-

gura 4.2), fissati rispettivamente a 11 e a 1, viene implementato l'algoritmo di *Sparse PCA* in modo da estrarre le matrici degli *scores* e dei *loadings*. Successivamente viene implementato il metodo Elbow per la scelta del numero di cluster: il risultato, mostrato in figura 4.7, ci consente di scegliere un valore pari a 8.

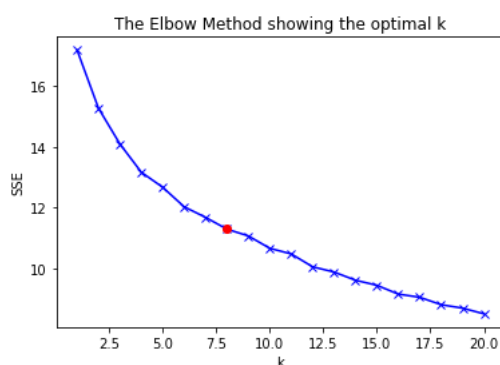


Figura 4.7: Metodo Elbow per la scelta del numero di cluster

Dalla matrice degli *scores* otteniamo i grafici mostrati in figura 4.8, 4.9 e 4.10, mentre dalla matrice dei *loadings* ricaviamo la rappresentazione a nube di parole raffigurata in 4.11.

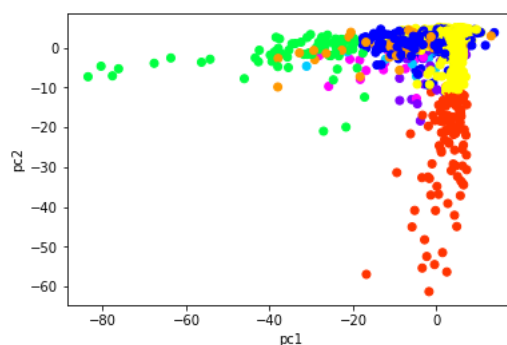


Figura 4.8: Rappresentazione dei documenti in un grafico bidimensionale

In questo caso i topic fondamentali della seconda collezione di dati testuali

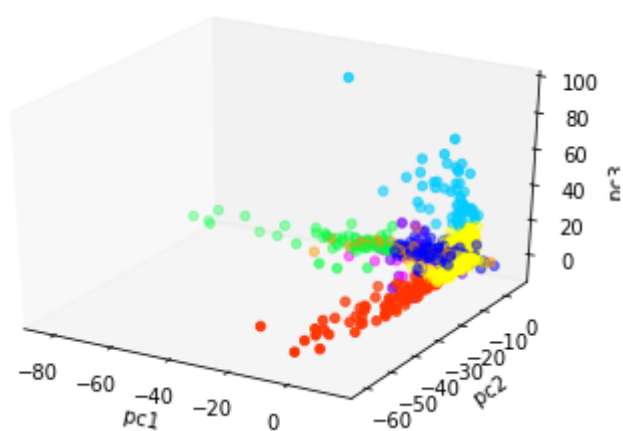


Figura 4.9: Rappresentazione dei documenti in un grafico tridimensionale

estratti da Wikipedia sono 8: matematica, politica, cucina, geografia, storia, musica, letteratura e astronomia. Anche questa volta l'algoritmo consente di individuare gruppi correlati e coesi di documenti testuali.

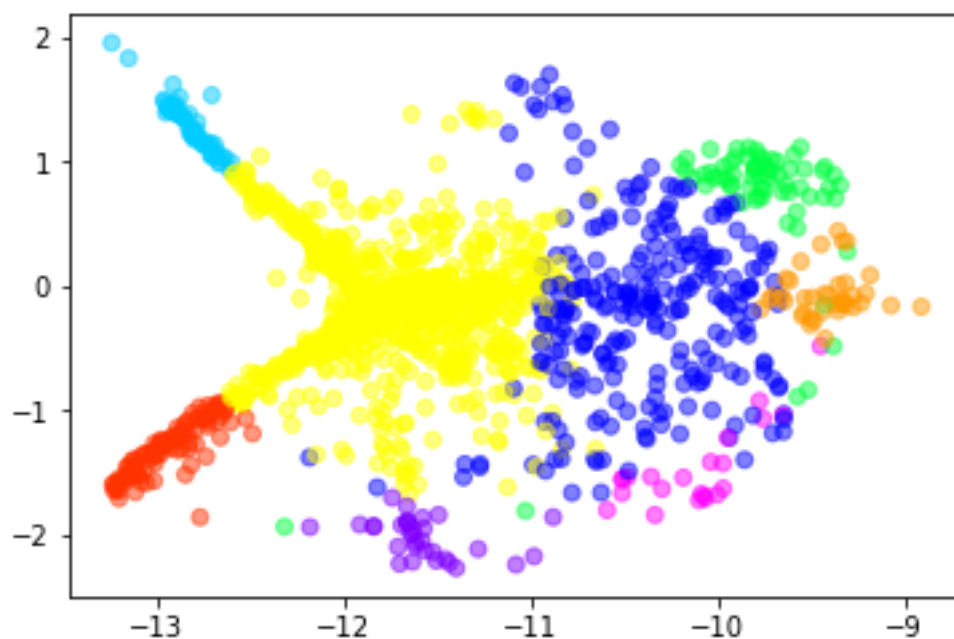


Figura 4.10: Rappresentazione t-SNE

Figura 4.11: Rappresentazione *word-cloud*

4.2 Collezione di Tweet

Spostiamo ora l'analisi su un altro caso di studio che presenta caratteristiche testuali diverse da quelle analizzate fin ora: nei paragrafi che seguono faremo riferimento a una raccolta di Tweet riguardanti eventi catastrofici avvenuti nel 2012 e nel 2013. Anche in questo caso, una volta svolte le varie opera-

zioni di *pre-processing* e calcolata la matrice $TF - IDF$, bisognerà scegliere opportuni valori per i parametri K e α . A tal proposito osserviamo i grafici mostrati in figura 4.12: selezioniamo un valore di K pari a 6, in modo da poter spiegare una percentuale di varianza aggiustata pari al 60%, e un α uguale a 1.75 (per questo valore la curva RMSE presenta un picco negativo, il che significa che si avrà una percentuale di varianza spiegata maggiore rispetto a quella che si otterrebbe fissando α a 0.5).

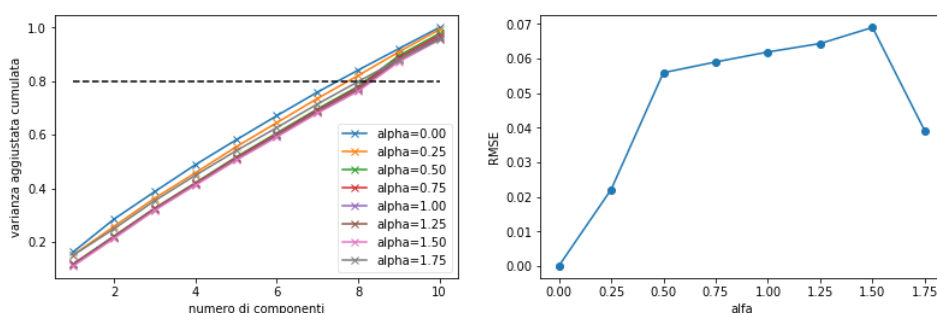


Figura 4.12: Scelta dei parametri K e α per una collezione di Tweet

Possiamo a questo punto implementare l'algoritmo di *Sparse PCA* impostando i parametri appena selezionati. Successivamente dovremo calcolare il numero di cluster che dovranno essere considerati nell'algoritmo di *K-Means Clustering*: il risultato del metodo Elbow è mostrato in figura 4.13 e ci consente di selezionare un valore pari a 3.

Di conseguenza implementiamo l'algoritmo di *K-Means Clustering* a partire dalla matrice degli *scores*, ottenuta dalla *Sparse PCA*, in modo da ricavare le rappresentazioni mostrate in figura 4.14 e 4.15.

Inoltre, anche in questo caso, possiamo rappresentare ogni topic per mezzo di una *word cloud*; in particolare osservando la figura 4.16 possiamo concludere che i tre argomenti principali della collezione sono i seguenti: l'attentato alla maratona di Boston avvenuto nel 2013, l'alluvione del 2013 verificatasi in Canada e l'uragano Sandy (2012).

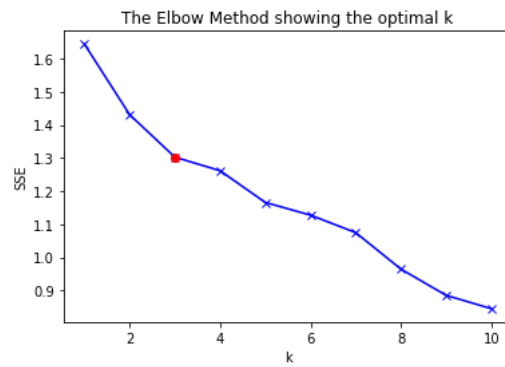


Figura 4.13: Metodo Elbow per la scelta del numero di cluster

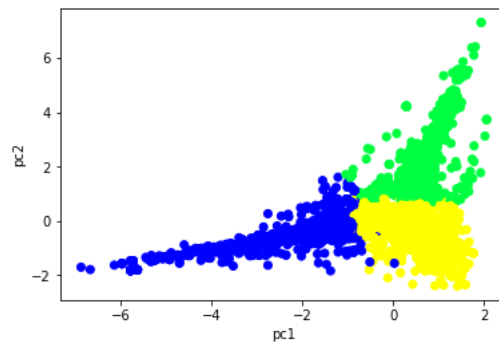


Figura 4.14: Rappresentazione dei documenti in un grafico bidimensionale

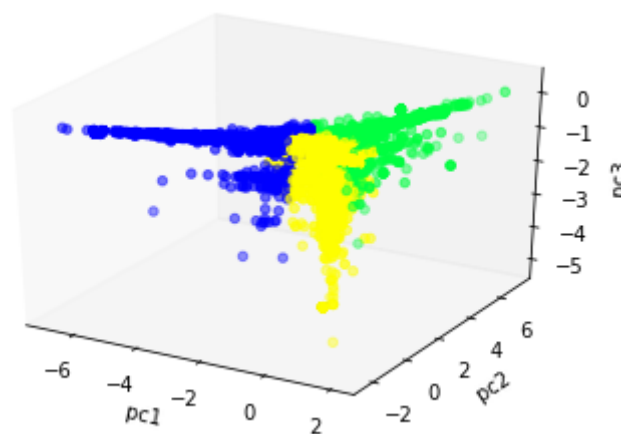


Figura 4.15: Rappresentazione dei documenti in un grafico tridimensionale

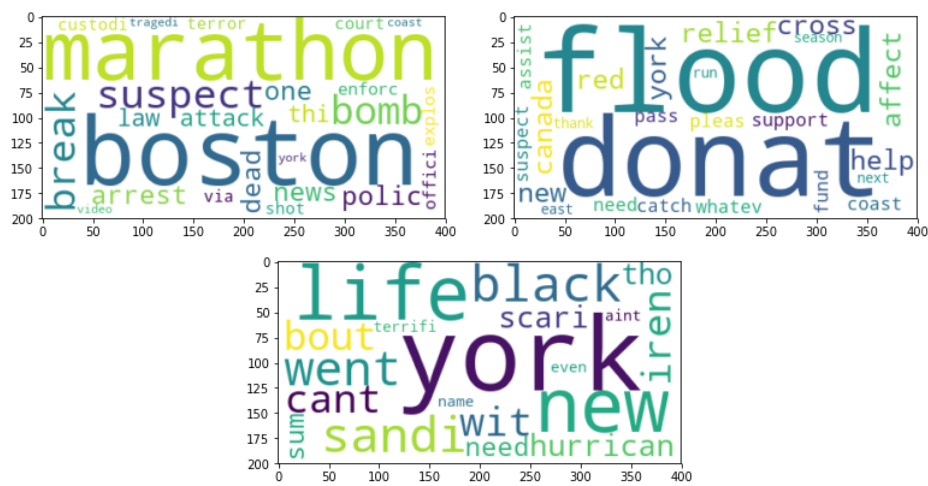


Figura 4.16: Rappresentazione *word-cloud*

Conclusione e sviluppi futuri

Questa tesi propone una strategia di selezione automatica che consente di semplificare il processo di analisi. I documenti presenti in una collezione di dati testuali vengono così raggruppati automaticamente in gruppi coerenti e ben separati attraverso una metodologia di *topic modeling*. La tecnica proposta in questo elaborato è l'analisi delle componenti principali sparse la quale, dati il numero di topic e il parametro di sparsità, ci consente di individuare documenti che appartengono allo stesso topic e per ogni topic ci fornisce una lista di parole che meglio lo descrivono. L'algoritmo è stato sviluppato in Python e convalidato su tre raccolte di dati reali, con caratteristiche e proprietà testuali diverse, tra cui due collezioni di dati testuali importati da Wikipedia e una collezione di Tweet. Dai risultati sperimentali ottenuti, il *tuning* dei parametri supporta in maniera automatica l'analista. L'algoritmo è in grado di raggruppare automaticamente i documenti presenti in una collezione di dati testuali in gruppi coerenti e ben separati e di fornire, per ogni topic, la lista dei termini che lo caratterizzano.

Infine presentiamo alcuni possibili sviluppi futuri di questo lavoro, tra cui:

- l'introduzione di nuovi pesi locali come, ad esempio, *Term-Frequency* (TF), *Logarithmic term frequency* (Log) e *Unitary* (Boolean), e globali, *Inverse Document Frequency* (IDF), *Entropy* (Entropy) e *Term-Frequency* (TF_{glob}). In questo modo si potrebbe pensare di usare matrici diverse rispetto alla matrice $TF - IDF$, date dalla combinazione di un peso locale e di uno globale, tra cui $TF - Entropy$, $Log - IDF$, $Log - Entropy$ e $Boolean - TF_{glob}$;

- rendere l'approccio scalabile su un framework *BigData* in grado di gestire una quantità illimitata di lavori simultanei;
- valutare tecniche alternative, come le reti neurali (*Deep Neural Network*, *DNN*), in modo da analizzarne i risultati e poterli così confrontare con quelli ottenuti dal modello proposto in questo elaborato.

Bibliografia

- [1] Stefano Proto, Evelina Di Corso, Francesco Ventura and Tania Cerquitelli: *Useful ToPIC: Self-tuning strategies to enhance Latent Dirichlet Allocation*, IEEE International Congress on Big Data (BigData Congress), 33-40, 2018.
- [2] Youwei Zhang, Laurent El Ghaoui: *Large-Scale Sparse Principal Component Analysis with Application to Text Data*, Advances in Neural Information Processing Systems, 532-539, 2011.
- [3] Giuseppe C. Calafiore, Laurent El Ghaoui, Alessandro Preziosi, Luigi Russo: *Topic analysis in news via sparse learning: a case study on the 2016 US presidential elections*, IFAC PapersOnLine 50-1 (2017) 13593–13598, 2017.
- [4] Vu Pham and Laurent El Ghaoui: *Robust sketching for multiple square-root LASSO problems*, Department of Electrical Engineering and Computer Sciences University of California, Berkeley, 2014.
- [5] Laurent El Ghaoui, Vu Pham, Guan-Cheng Li, Viet-An Duong, Ashok Srivastava and Kanishka Bhaduri: *Understanding Large Text Corpora via Sparse Machine Learning*, Statistical Analysis and Data Mining, The ASA Data Science Journal 6 (3), 221-242 , 2013.
- [6] Blei et al.: *Latent Dirichlet Allocation*, 2003.
- [7] Zou et al.: *Sparse Principal Component Analysis*, 2006.

- [8] Christophe Croux, Peter Filzmoser e Heinrich Fritz: *Robust Sparse Principal Component Analysis*, 2013.
- [9] Vines: *Simple principal components*, 2000.
- [10] Cadima and Jolliffe: *Variable selection and the interpretation of principal subspaces*, 2016.
- [11] A. Olteanu, C. Castillo, F. Diaz, S. Vieweg. 2014. CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises. In Proceedings of the AAAI Conference on Weblogs and Social Media (ICWSM'14). AAAI Press, Ann Arbor, MI, USA.